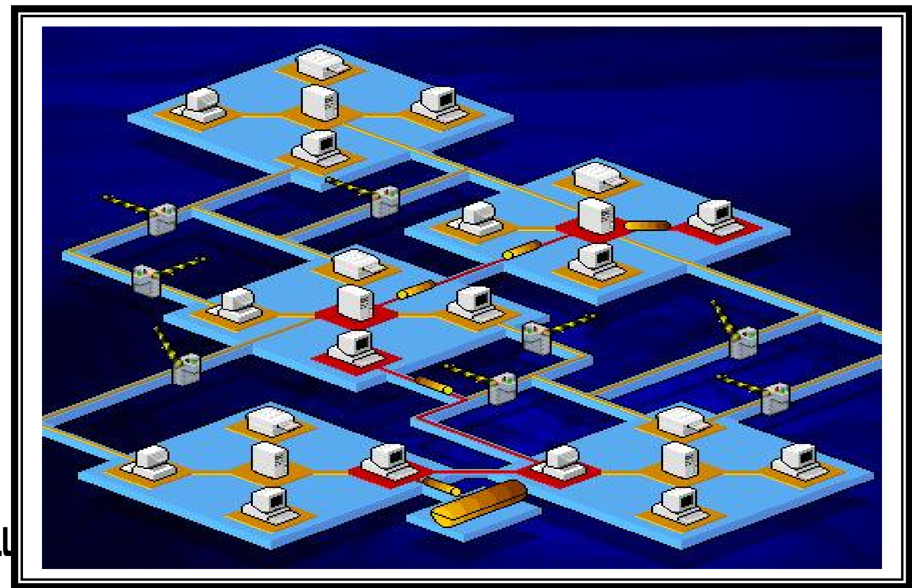
كما ذكرنا سابقاً فأن انظمة الحاسب الألي تتكون من مجموعه من الموارد (الطابعات ، الأقراص ، ... ، الخ) ، وكما انه يمكن لأي عمليه استخدام واحد أو أكثر من تلك الموارد لأنجاز عملها .

فمثلاً اذا كان لدينا عمليتان C & D وتريدان قراءة ملف عبر الماسحه الضوئيه ثم تقومان بعد ذلك بنسخه الي قرص مدمج .

أولاً تطلب العملية C أستخدام الماسحه الضوئيه ويتم لها ذلك ، وبعد ذلك تطلب العملية D أستخدام ناسخه الأقراص المدمجه ويتم لها ذلك أيضاً ثم تطلب العمليه C ناسخه الأقراص المدمجه لكن طلبها يرفض حتي تتخلي العملية D عن ناسخه الأقراص المدمجه .

وبنفس الطريقه تطلب العمليه D استخدام الماسحه الضوئيه ولكن للأسف يرفض طلبها حتي تتخلي العمليه C عن الماسحه الضوئيه ، وبذلك ستبقي كلتا العمليتان في حالة توقف للأبد وهذا مايسمي بالأستعصاء DEADLOCK .

ونفس الطريقة يمكن ان يحدث الأستعصاء DEADLOCK علي المستوي المادي (الأجهزة) ، فمثلاً اذا كان لدينا مجموعه من الحاسبات متصله مع بعضها البعض عن طريق شبكه محليه وكان لدينا مجموعه من الأجهزة (طابعات ، ماسحات ضوئيه ، ... ، الخ) تستخدم كموارد مشتركه لجميع الحاسبات فيمكن ان يحدث الأستعصاء DEADLOCK حول ثلاثه أو اربعه من تلك الموارد من قبل عده مستخدمين



أعداد :-
أ. الهادي سلي... ...اسوب

مارس ٢٠٠٨

وكما أنه يمكن ان يقع الأستعصاء علي مستوي قواعد البيانات وعلي سبيل المثال اذا أفترضنا انه يوجد لدينا نظام قواعد بيانات وكانت هنالك عمليتان تريدان التعديل في قاعدة البيانات وبأفتراض ان العمليتان هما P1 & P2 والسجل المراد التعديل فيه هو R2 & R1 فإذا قامت العمليه P1 بقفل السجل R1 لأجراء التعديل فيه وفي نفس الوقت قامت العمليه P2 بقفل السجل R2 لنفس الشئ ، ثم بعد ذلك ارادت كلتا العمليتان P1 & P2 قفل سجل الأخري وفي هذه الحاله سوف يحدث لدينا ما يسمي بالأستعصاء DEADLOCK .

عموماً يمكن ان يحدث الأستعصاء علي كل مستويات الموارد (المادية ، البرمجيه) وبذلك يمكن ان نعرف المورد بإنه أي شئ يمكن أن تستخدمه عمليه ما في لحظه من الزمن .

أنواع الموارد

وكما ذكرنا سابقاً فإن المورد هو أي شئ يمكن لعمليه ما استخدامه في لحظه من الزمن ، ومن هذا التعريف يمكننا ان نقسم الموارد الي نوعين هما

١. موارد قابله للأستيلاء Preemptable Resource .
٢. موارد غير قابله للأستبلاء NON Preemptable Resource .

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

١. موارد قابله للأستيلاء Preemptable Resource

يمكن ان تعرف الموارد القابله للأستيلاء بإنها تلك الموارد التي يمكن ان تنتزع من العمليه دون ان يسبب ذلك أي مشكله ومن أمثلة الموارد القابله للأستيلاء الذاكرة .

فمثلاً اذا كان لدينا ذاكرة متاحه بحجم 16 MB وكانت هنالك عمليتان P1 & P2 كل منهما تحتاج 16 MB من الذاكرة لطباعه ملف معين علي الطابعه .
تقوم أولاً العمليه P1 بعملية الطباعه وذلك بحساب القيم التي تريد طباعتها وقبل ان تنجز عملها يتم أستبدالها بالعمليه P2 وذلك لتجاوزها الزمن المخصص لها ، وتطلب العمليه P2 الطابعه لطباعة ملفها ولكن طلبها يرفض لأن الطابعه في حوزت العملية P1 .
وظاهرياً يبدو لنا أنه سوف يحدث أستعصاء وذلك لأن العملية P1 تملك الطابعة والعملية P2 تملك الذاكرة وكل منهما لا تستطيع متابعة عملها دون الحصول علي المورد الذي بحوزت الأخري .
ولكن يمكننا ان نستولي علي الذاكرة من العملية P2 واعطاءها للعملية P1 بدون حدوث أي مشكلة وذلك بإدخال العملية P1 الي الذاكرة مرة أخري لأكمال عملها وتحرر الطابعه وبذلك نكون قد منعنا حدوث الأستعصاء DEADLOCK .

٢. موارد غير قابله للأستبلاء NON Preemptable Resource

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

وايضاً يمكن ان تعرف الموارد الغير قابله للأستيلاء علي انها تلك الموارد التي لا يمكن انتزاعها من صاحبها دون حدوث أي مشكلة ، ومن أمثلة الموارد الغير قابلة للأستيلاء ناسخة الأقراص المدمجة .

فعموما يحدث الأستعصاء DEADLOCK في الموارد الغير قابلة للأستيلاء ، أما الأستعصاء الظاهري والذي يحدث في الموارد القابلة للأستيلاء فيمكن حله وذلك بإعادة ترتيب الموارد بإخذها من عملية واعطائها لعملية أخري .

الحصول علي الموارد

يتم الحصول علي المورد وذلك بتنفيذ عدة خطوات وهي

١. طلب العملية للمورد .
٢. أستخدام العملية للمورد .
٣. تحرير العملية للمورد .

وبعض انظمة التشغيل تقوم بإدارة المورد وذلك بربط كل مورد بمسير Semaphore وتعطي جميع المسيرات القيمه ١ مبدئياً ثم بعد ذلك تقوم بتحقيق الثلاث خطوات السابقة مع كل عملية .

فتنفذ الأستدعاء Down علي المسير من اجل الحصول علي المورد المحدد ثم تنفيذ الأستدعاء USE لأستخدام المورد وعند الأنتهاء تقوم بتنفيذ الأستدعاء Up من اجل تحرير ذلك المورد .

```
Typedef  Int Semaphore
Semaphore     Resource _ A  ;
Void   Process _ P ( Void ) {
ـ  Down ( & Resource _ A ); // A مناداة الدالة Down لطلب المورد
```


أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

_ Use _Resource _ A ( ); // A لاستخدام المورد Use مناداة الدالة

_ Up (& Resource _ A); // A لتحرير المورد Up مناداة الدالة

}

(a) أستخدام مسير لحماية مورد واحد

Typedef Int Semaphore
Semaphore Resource _ A ;
Semaphore Resource _ B ;
Void Process _ P (Void) {

_ Down (& Resource _ A); // A لطلب المورد Down مناداة الدالة

Down (& Resource _ B); // B لطلب المورد Down مناداة الدالة

_ Use _Resource _ A ( ); // A لاستخدام المورد Use مناداة الدالة

_ Use _Resource _ B ( ); // B لاستخدام المورد Use مناداة الدالة

_ Up (& Resource _ A); // A لتحرير المورد Up مناداة الدالة

_ Up (& Resource _ B); // B لتحرير المورد Up مناداة الدالة

}

(b) أستخدام مسير لحماية موردين

كما لاحظنا في الشكل (b) فغالباً ما تحتاج العملية الي موردين أو أكثر ولكن ليس هنالك أي مشكلة طالما ان هنالك عملية واحدة ولا توجد تنافس للحصول علي تلك الموارد .
أما اذا كان لدينا عمليتان أو أكثر ونفترض ان لدينا العمليتان P1 & P2 وكان لدينا موردين A & B في هذه الحالة يكون لدينا أحتمالان .

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

أما ان تطلب كل من العمليتان P1 & P2 الموردين بنفس الترتيب دون أي مشاكل وذلك لأن أحدي العمليتان سوف تحصل علي المورد الأول قبل الأخري وبنفس الطريقه سوف تحصل علي المورد الثاني ، وفي هذه الحاله ستتوقف العملية الأخري ريثما يصبح المورد المطلوب متوفراً .

```
Typedef  Int Semaphore

Semaphore     Resource _ A  ;
Semaphore     Resource _ B  ;
Void   Process _ P1 ( Void ) {
- Down ( & Resource _ A );
  Down ( & Resource _ B );
- Use _Resource _ A ( );
- Use _Resource _ B ( );
- Up ( & Resource _ A );
- Up ( & Resource _ B );
}
```

```
Typedef  Int Semaphore

Semaphore     Resource _ A  ;
Semaphore     Resource _ B  ;
Void   Process _ P2 ( Void ) {
- Down ( & Resource _ A );
  Down ( & Resource _ B );
- Use _Resource _ A ( );
- Use _Resource _ B ( );
- Up ( & Resource _ A );
- Up ( & Resource _ B );
}
```

(a) طلب العمليتان P1 & P2 الموردين بنفس الترتيب

أما الأحتمال الثاني وهو أن تقومان بطلب الموردين بترتيب مختلف وهنا سوف يحدث مشكلة وهي ان كل عملية سوف تكون حصلت علي مورد واحد وتحتاج للمورد الذي بحوزة

أعداد :-

أ. الهادي سليمان الهادي   جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

الأخري مما يسبب اعاقه للعمليتان فستتوقف كل منهما عندما تحاول الحصول علي المورد الأخر وتصبح كل منهما غير قابلة للتنفيذ ابداً ويظهر لدينا الأستعصاء DEADLOCK .

Typedef Int Semaphore Semaphore Resource _ A ; Semaphore Resource _ B ; Void Process _ P2 (Void) { _ Down (& Resource _ B); Down (& Resource _ A); _ Use _Resource _ B ( ); _ Use _Resource _ A ( ); _ Up (& Resource _ B); _ Up (& Resource _ A); }	Typedef Int Semaphore Semaphore Resource _ A ; Semaphore Resource _ B ; Void Process _ P1 (Void) { _ Down (& Resource _ A); Down (& Resource _ B); _ Use _Resource _ A ( ); _ Use _Resource _ B ( ); _ Up (& Resource _ A); _ Up (& Resource _ B); }

(b) طلب العمليتان P1 & P2 الموردين بترتيب مختلف

من هذا كله يمك　　　كانت كل عملية من هذه المجموعة تحتاج الي حدوث حدث لا يمكن تحقيقة الا من قبل عملية أخـري من هذه المجموعة نفسها .

وفي أغلب الاحيان يكون ذلك الحدث هو عبارة عن تحرير مورد تمتلكه عمليه أخري حالياً وبالتالي لا يمكن تحرير أي عمليه مستعصيه ويبقي هذا الوضع مستمراً الي ما لا نهاية .

شروط الأستعصاء

قدم العالم كوفمان عام ١٩٧١ أربعه شروط أساسية يجب توفرها لكي نقول أن هذه العملية مستعصيه وهي علي النحو التالي :-

١) شرط منع التبادل

هو أن يكون كل مورد مرتبطاً أرتباطاً حصرياً بعملية واحدة .

٢) شرط الأحتجاز والأنتظار

هو ان تكون كل عمليه قد أمتلكت مورداً معيناً ولكنها تحتاج لمورد أخر لأنجاز عملها .

٣) شرط عدم الأستيلاء

هو ان لا يحدث أنتزاع للمورد من عمليه معينه قسرياً وانما تقوم العملية نفسها بعد أكتمال عملها بتحرير ذلك المورد .

٤) شرط الأنتظار الدائري

هو ان يكون هنالك سلسلة دائريه من العمليات كل واحده تحتاج لمورداً تمتلكه العمليه التي تليها في السلسلة .

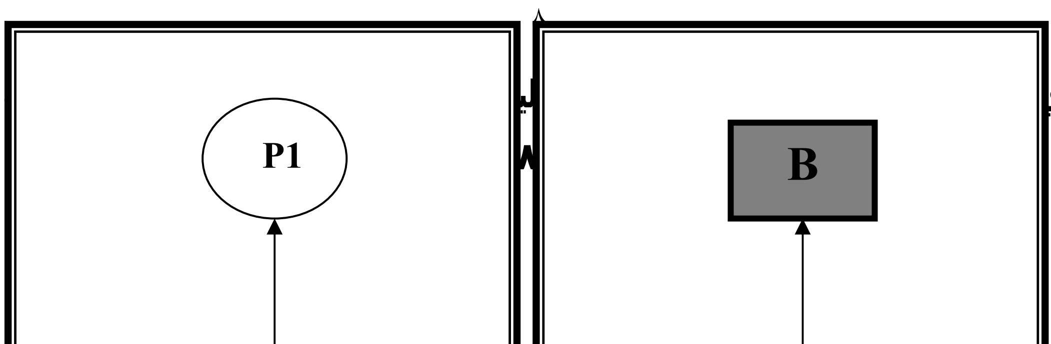
نمذجة الأستعصاءات

قام العالم هولت عام ١٩٧٢ بشرح كيف يمكننا نمذجة تلك الشروط الأربعه وذلك بإستخدام ما يعرف بالغرافات الموجهة .

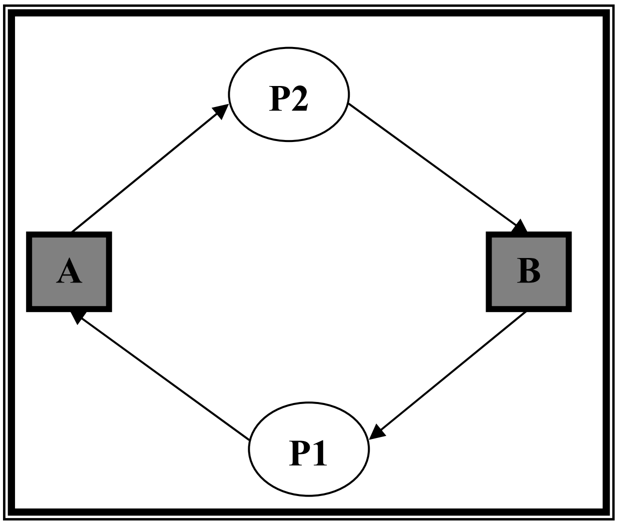
حيث قام بتمثيل المورد بعقده تحمل شكل المربع بينما مثل العملية بعقده تحمل شكل الدائرة ، أما القوس الذي يبدأ من عقده المورد (المربع) ويتجه الي عقده العملية (الدائرة) فيعني هذا ان ذلك المورد قد طلبته تلك العملية وحازت عليه .

واما القوس الذي يبدأ من عقده عمليه (دائرة) ويتجه الي عقده مورد (مربع) فيعني ان العمليه قد طلبت ذلك المورد ولم تمتلكه بعد .

أعداد :-
أ. الهادي



علي سبيل المثال يمكن تمثيل أستعصاء بسيط حدث بين عمليتان P1 & P2 حيث كانت العملية P1 تحتاج الي المورد A والذي كان بحوزة العملية P2 ، بينما كانت العملية P2 تحتاج الي المورد B والذي كان بحوزة العملية P1 .



أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

مثال أخر :-

اذا كانت لدينا العمليات P1 & P2 & P3 وكانت لدينا الموارد التالية A & B & C وكانت حالة الموارد المستخدمه حالياً والموارد المطلوبه كما يلي

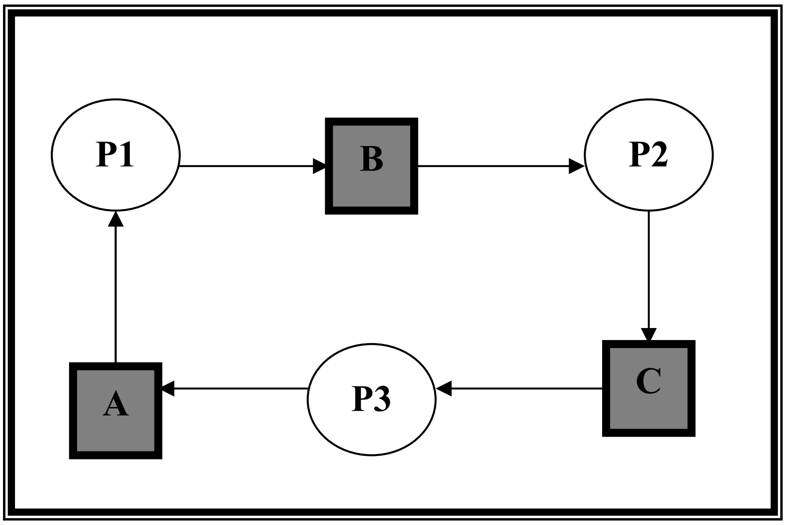
١) العملية P1 تمتاك المورد A وتحتاج الي المورد B .

٢) العملية P2 تمتاك المورد B وتحتاج الي المورد C .

٣) العملية P3 تمتاك المورد C وتحتاج الي المورد A .

مستخدماً الغرافات الموجهة لتمثيل ذلك الأستعصاء .

الحل



أعداد :-

أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

والان وبعد ماتحدثنا عن الأستعصاء وماهي الشروط التي يجب توفرها ليحدث وكيف يتم نمذجة نقوم بعرض الطرق التي تمكننا من التعامل مع الأستعصاءات ، وبشكل عام توجد أربعة خطط عمل من أجل التعامل مع الأستعصاءات وهي :-

١) الخطة الأولي

فكرة هذه الطريقة أن نقوم بتجاهل المشكلة تماماً . ومن أمثلة هذه الطريقة خوارزمية النعامة (دس رأسك في الرمل) وهي أبسط الطرق بحيث تتظاهر بإنه لاتوجد أي مشكلة أطلاقاً (وفي الواقع النعامة لا تقوم بذلك بل أنها تستطيع الركض بسرعه ٦٠ كلم / ساعه بالأضافة الي أنها يمكنها قتل عدوها برفسه واحده منها) .
ويختلف الناس حول طرق التعامل مع هذه الطريقة ، فمثلاً الرياضيون يجدونها غير مقبوله أطلاقاً ويقولون يجب منع الأستعصاءات مهما كان الثمن .
بينما المهندسون يقولون اذا كانت الأستعصاءات تحدث بمعدل أستعصاء كل خمس سنوات فيما ينهار النظام بسبب أسباب أخري مثل فشل في المكونات أو أخطاء في شفرة النظام بمعدل مره كل أسبوع فأنهم لا يرغبون ببذل جهودهم والتضحيه بالاداء من أجل أزالة الأستعصاءات .

٢) الخطة الثانية

فكرة هذه الطريقة تتلخص في ترك الأستعصاءات ان تحدث ثم بعد ذلك نقوم بإكتشافها واتخاذ الأجراءات المناسبة لحلها .

كشف الأستعصاءات

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

نفترض ان لدينا نظام يحتوي علي ماسحة ضوئية واحدة وناسخة أقراص واحدة وراسمة واحدة أي ان هنالك مورد واحد من كل صنف وكان النظام يحتوي علي سبع عمليات من P1 الي P7 وسته موارد من R1 الي R6 ، وتعطي حالة الموارد المستخدمة حالياً والموارد المطلوبه علي النحو التالي :-

١) العملية P1 تمتلك R1 لكنها تحتاج R2 .

٢) العملية P2 لا تمتلك شئ لكنها تحتاج R3 .

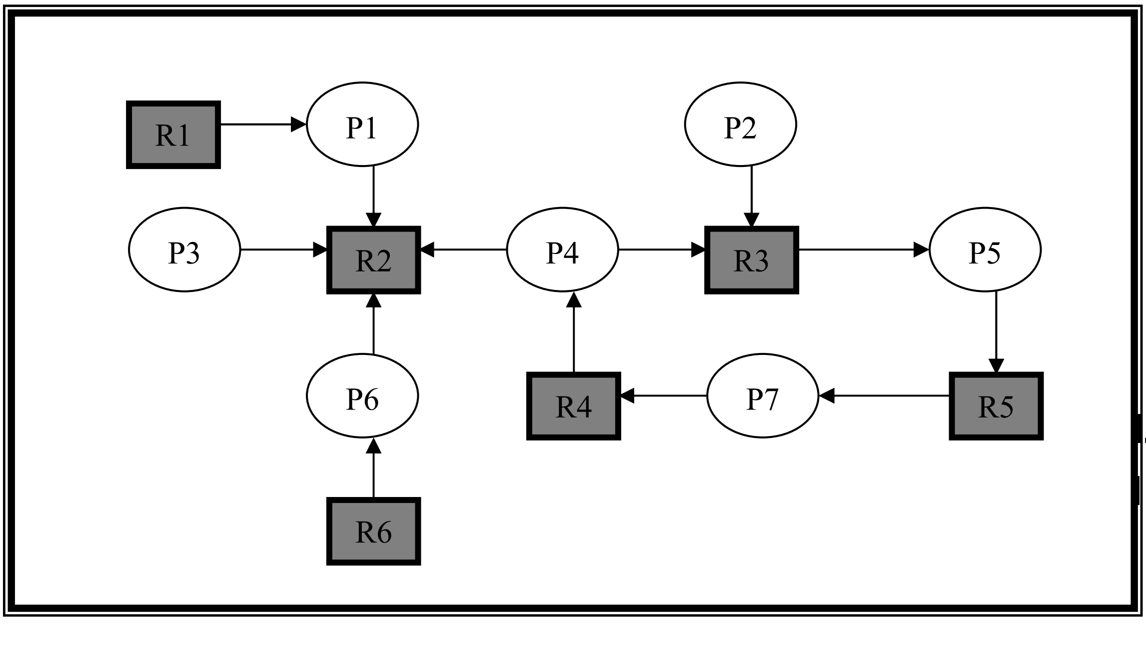
٣) العملية P3 لا تمتلك شئ لكنها تحتاج R2 .

٤) العملية P4 تمتلك R4 لكنها تحتاج R2 & R3 .

٥) العملية P5 تمتلك R3 لكنها تحتاج R5 .

٦) العملية P6 تمتلك R6 لكنها تحتاج R2 .

٧) العملية P7 تمتلك R5 لكنها تحتاج R4 .



بالرغم من أننا يمكننا معرفة العمليات المستعصية من خلال غراف بسيط الا اننا نحتاج الي خوارزمية رسمية لكشف الأستعصاءات وذلك لاستخدامها في الأنظمة الفعليه .
كما أنه يوجد العديد من الخوارزميات المعروفه لكشف الحلقات في الغرافات الموجهة من بين هذه الخوارزميات سوف ندرس خوارزمية بسيطه والتي تقوم أولاً بفحص الغراف وتنتهي أما عند وجود حلقه أو تبرهن علي عدم وجود حلقه ، كما انها تستخدم بنيه بيانات عبارة عن قائمة من العقد وتقوم الخوارزمية بتعليم الأقواس الخارجه من كل عقدة في الغراف للدلاله علي انها قد تم فحصها وذلك لتجنب الفحص المتكرر ، وتتبع الخوارزمية الخطوات التالية :-

١. أجعل N هي عقدة البداية (نختار عشوائياً أي عقدة من الغراف ولتكن N) .
٢. أجعل القائمة ولتكن L فارغه .
٣. أجعل جميع الأقواس الخارجه من الغراف غير معلمه .
٤. أضف العقدة N الي نهاية القائمة L .
٥. أختبر العقدة N هل هي موجودة مرتين في القائمة أذا كانت نعم فإن الغراف يحوي حلقه وتنتهي الخوارزمية والا أذهب الي الخطوة رقم ٦ .
٦. أختبر الأقواس الخارجه من العقدة N هل هنالك أي قوس غير معلم فإذا كانت الأجابة نعم أذهب الي الخطوة رقم ٧ والا أذهب الي الخطوة رقم ٨ .
٧. أختار أحد الأقواس عشوائياً وعلمه ثم أجعل العقدة N هي العقدة التي يشير اليها ذلك القوس ثم أذهب الي الخطوة رقم ٤ .

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

٨. وصلنا الي نهاية ميته لذلك سوف نقوم بالرجوع إلى العقدة السابقة للعقدة N ونجعلها N
٩. نختبر العقدة N هل هي عقدة البداية فإذا كانت الأجابة بنعم فإننا قد وصلنا الي النهاية ولا توجد أي حلقة والا أذهب الي ٤ .

والأن سوف نطبق تلك الخوارزمية علي المسأله السابقة ، أولاً نقوم بترتيب العقد عشوائياً ولتكن

R1 , P1 , P2 , P3 , R2 , P4 , R3 , P5 , P6 , R4 , R6 , P7 , R5 .

فإذا وجدت الخوارزمية حلقة سوف تنتهي ، والأن سنبدأ خطوات الخوارزمية :-

نجعل L قائمة فارغه ثم نختار عقدة البداية ولتكن R1 (العقدة الأولي من جهة اليسار في الترتيب) ، نضيف R1 الي القائمة L ثم ننتقل الي العقدة التالية لها (العقدة التي يتجه قوس من R1 اليها) ، وهي P1 ونقوم بإضافتها الي القائمة فتصبح L = [R1 , P1]

ثم بعد ذلك ننتقل الي R2 ونضيفها الي القائمة فتصبح L = [R1 , P1 , R2]

ونلاحظ انه لا يوجد أي قوس خارج منها (R2) فهي نهاية ميته لذلك نعود الي العقدة P1 وبما انها ليس لها قوس خارج منها أخر سوف نعود الي R1 وبذلك يكون قد تم فحص العقدة R1 ولا يوجد لديها أي حلقة .

والأن سوف نعيد الخوارزمية بإستخدام العقدة P1 (وهي العقدة الثانية في الترتيب العشوائي الأولي) .

ثم نجعل L فارغه ونضيف P1 الي القائمة ولكن البحث سوف ينتهي بسرعه لوجود النهاية الميته السابقة ، لذلك سوف نبدأ الخوارزمية من عقدة أخري وهي P2 (العقدة الثالثة في الترتيب) ، ونتابع الأقواس الخارجه من P2 حتي نصل الي العقدة P4 وعندها تكون القائمة هي :-

L = [P2 , R3 , P5 , R5 , P7 , R4 , P4]

ثم نختار أي قوس خارج من P4 عشوائياً ، فإذا قمنا بأختيار القوس المتجه الي العقدة R2 فسوف نصل الي نهاية ميته لذلك سوف نعود الي P4 ونختار القوس المتجه الي العقدة R3 وبذلك تصبح القائمة علي النحو التالي :-

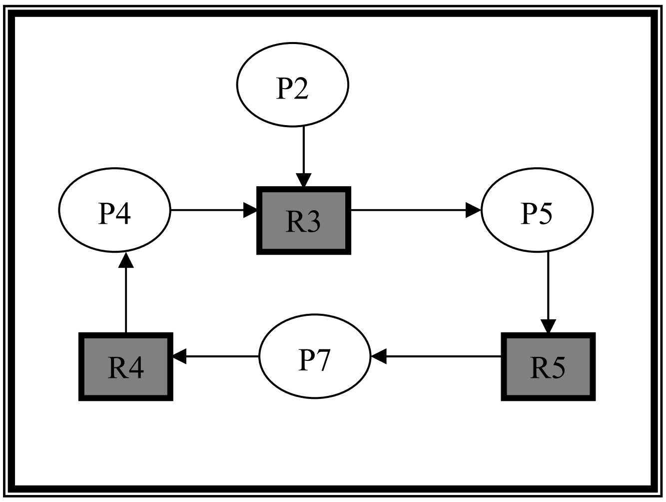
أعداد :-

أ. الهادي سليمان الهادي　　جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

L = [P2 , R3 , P5 , R5 , P7 , R4 , P4 , R3]

وفي هذه اللحظه نكتشف أن العقدة R3 قد تكررت وبذلك يكون هنالك حلقه وتنتهي الخوارزمية بإكتشاف الحلقه التاليه :-



وبعد ما تعرفنا علي الأستعصاء (بوجود حلقه في الغراف) سوف نقوم بدراسه عدة حلول لحل ذلك الأستعصاء وهي :-

١. التخلص القسري من الأستعصاء

فكرة هذا الحل هو أخذ المورد المسبب للأستعصاء من العملية وأعطائه لعملية أخري دون حدوث أي مشكلة للعملية الأولي ثم أعادته اليها بعد أنتهاء العمليه الثانيه منه ،

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

ولكن هذا الحل عادة ما يكون صعباً أو مستحيلاً وذلك لانه يعتمد بشكل كبير علي طبيعية المورد والعمليات التي يمكن أعادة الموارد اليها بسهولة دون حدوث أي مشكلة .

٢. التخاص من الأستعصاء عن طريق التراجع

فكرة هذا الحل ان يكون هنالك نقاط أختبار دوريه لكل عمليه وتحتوي تلك النقاط علي الموارد المخصصه لهذه العمليه عند كل نقطه .
وبذلك يكون لدينا تسلسل تراكمي من نقاط الأختبار فعند حدوث الأستعصاء يكون من السهل لنا معرفه المورد المسبب ثم بعد ذلك نقوم بإعادة العملية التي تملك ذلك المورد الي النقطة التي تسبق عمليه حجز المورد ، واذا أرادت العمليه التي تراجعت الحصول علي المورد مره أخري فإنها ستضطر للأنتظار حتي يصبح متاحاً مره أخري .

٣. التخاص من الأستعصاء عن طريق قتل العملية

فكرة هذه الطريقة تتلخص في قتل أحدي العمليات المشتركه بالحلقه وبذلك يمكن لجميع العمليات الأخري متابعة عملها ، واذا لم يفد ذلك يمكن أختيار عمليه أخري من بين العمليات المشتركه وقتلها .
لذلك يجب المراعاه عند أختيار العملية المراد قتلها بعناية بحيث تكون تمتلك الموارد التي تحتاجها عملية أو اكثر في الحلقه .
كما يفضل بقدر الأمكان قتل عمليه يمكن أعادة تشغيلها مره أخري دون حدوث أي أثار جانبيه ضارة فعلي سبيل المثال يمكن أعادة عمليه ترجمه من البداية بعد قتلها لانه كل ما

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

قامت به قبل القتل عباره عن قراءة ملف المصدر وأنتاج ملف الهدف وبذلك يكون التشغيل الأول ليس له أي علاقه بالتشغيل الثاني .
ولكن لا يمكن قتل عمليه تريد أضافة مثلا الرقم ١ الي سجل معين في قاعدة البيانات وذلك لأن قتلها ثم تشغيلها مره أخري يؤدي الي أضافة الرقم ٢ الي السجل المعني (أضافة الرقم ١ قبل قتلها ثم أضافته مرة أخري عند تشغيلها مرة أخري) وهي نتيجة غير صحيحه .

٣) الخطه الثالثه

معظم العمليات تطلب الموارد واحداً تلو الأخر لذلك يجب علي النظام معرفة هل منح المورد لعمليه معينه أمن (لا يسبب أستعصاء) أم لا ، فإذا كان أمناً يمنحه لها أي أن هذه الخطه تهدف الي تجنب الأستعصاء قبل حدوثه ، كما انه هنالك عدة خوارزميات نستطيع من خلالها تجنب الأستعصاءات وذلك من خلال تخصيص المورد بحرص شديد .
وسوف ندرس الأن واحدة من هذه الخوارزميات وهي :-

خوارزمية المصرفي Banker

قام العالم ديكاسترا في عام ١٩٦٥ بكتابه خوارزمية نستطيع من خلالها تجنب الأستعصاءات قبل حدوثها وعرفها بخوارزمية المصرفي Banker وهي في الحقيقة عبارة عن توسيع لخوارزميات أكتشاف الأستعصاءات كما أنها تعتمد في عملها علي الطريقه التي يتعامل بها موظف المصرف مع مجموعه من زبائنه الذين يمتلكون حسابات أقتراض في المصرف .

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

بأفتراض أن لدينا الزبائن A , B , C , D كل منهم لديه عدداً من وحدات الأقتراض كما ان المصرفي يظن ان الزبائن لن يحتاجون للحد الأعلي من قروضهم في دفعه واحدة لذلك فقد حجز ١٠ وحدات فقط لخدمتهم مع العلم بإن مجموعه قروض الزبائن مجتمعه تساوي ٢٢ وحدة .

نلاحظ أن العالم قام بنمذجة العمليات والموارد المطلوبه والمتاحة ونظام التشغيل وذلك بتمثيل العمليات بالزبائن بينما قام بتمثيل الموارد المتاحة والمطلوبه بوحدات القروض وكما أنه مثل نظام التشغيل الذي يدير كل هذه الموارد والعمليات بالمصرفي .

وكل زبون يقوم بإدارة عمله بشكل خاص وذلك بطلب بعض القروض من حين الي أخر فمثلاً أذا كانت الحاله الأبتدائية للمصرفي والزبائن A , B , C , D كما هو موضح في الشكل أدناه

الوحدات التي يمتلكها كل زبون

الحد الأعلي للقروض لكل زبون

A	0	6
B	0	5
C	0	4
D	0	7

الوحدات المتاحه لدي المصرفي = ١٠ وحدات

واذا أفترضنا ان في لحظه ما قام الزبائن بطلب الوحدات التالية من المصرفي

أعداد :-

أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

A = 1 , B = 1 , C = 2 , D = 4

يكون لدينا الشكل التالي :-

A	١	6
B	١	5
C	٢	4
D	٤	7

الوحدات المتاحه لدي المصرفي = ٢ وحدة

وهذه الحاله يمكن أعتبارها حاله أمنه لأن لدي المصرفي وحدتين وبذلك يمكن تجميد كل العمليات ما عدا العمليه C مما يسمح لـ C بأتمام عملها وتحرير جميع الوحدات الأربعه كما في الشكل التالي :-

A	١	6
B	١	5
C	٠	4
D	٤	7

الوحدات المتاحه لدي المصرفي = ٤ وحدات

وبذلك يكون لدي المصرفي ٤ وحدات متاحه ويستطيع بعد ذلك اعطاء الزبون D أو B الوحدات اللازمة له لأكمال عمله وهكذا .
أما اذا قمنا بمنح وحدة واحدة أخري للزبون B وكما موضح في الشكل أدناه

A	١	6
B	٢	5
C	٢	4
D	٤	7

الوحدات المتاحه لدي المصرفي = ١ وحدة

فينتج لدينا حاله غير أمنه وذلك لأن المصرفي لأ يملك سوي وحدة واحدة وهي غير كافيه لتلبيه طلب أحد الزبائن الأربعه مع الرقم من أن الزبون يمكن أن لا يحتاج الي الحد الأعلي من القروض دفعه واحدة ولكن يجب علي المصرفي أن لا يعتمد علي ذلك .

فعموماً تقوم خوارزمية المصرفي بدراسة كل طلب لدي وقوعه وتختبر هل منحه سوف يؤدي الي حاله أمنه فإذا كان كذلك يتم منح الطلب والأ سوف يؤجل هذا الطلب الي ما بعد . وتكون الحالة أمنه اذا كانت لدي المصرفي وحدات كافية لتلبيه أحد الزبائن وعندها يفترض أن هذه الوحدات قد تم سدادها وبعد ذلك يفحص الزبون الأقرب الي الحد الأعلي من الوحدات وهكذا ، فإذا كانت جميع القروض يمكن سدادها في النهاية نقول أن الحاله أمنه وبعد ذلك يتم منح الطلب الأبتدئي .

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

٤) الخطه الرابعه

وفي الحقيقه لا يمكن عملياً تجنب الأستعصاءات وذلك بسبب الحاجه لوجود معلومات كافية لجميع الطلبات المستقبليه والتي لا يمكن التنبؤ بها لذلك كانت فكرة الخطه الرابعه تتلخص في أننا يمكننا منع الأستعصاءات وذلك بعدم تحقق أحد الشروط الأربعه علي الأقل وبذلك يصبح الأستعصاء مستحيلاً من الناحيه البنويه .
والأن سوف ندرس كل شرط علي حد ومحاولة عدم تحقيقه وبذلك يكون قد منعنا الأستعصاء .

١. مهاجمة شرط منع التبادل

يمكن مهاجمة الشرط الأول وذلك بعدم تخصيص أي مورد بشكل حصري علي عمليه واحده ولكن هذا يمكن أن يحدث أخطاء فعلي سبيل المثال اذا قامت عمليتان بالكتابه الي الطابعه في نفس الوقت سيؤدي الي طباعة نص غير مفهوم .
ولكن يمكننا أستخدام جدولة الطباعة (Spooling) ، وبذلك يمكن لعدة عمليات من أستخدام الطابعه في نفس الوقت دون حدوث أي مشاكل لأن العمليه الوحيده التي تطلب الطابعه فعلياً هي مشغل الطابعه (Printer Demon) ، وبما أن ذلك

المشغل لا يستخدم أي موارد أخري سوي الطابعه فإننا بذلك نكون قد تخلصنا من استعصاءات الطابعه ، ولكن لا يمكن أستخدام هذه الطريقه مع جميع الأجهزة .

٢. مهاجمة شرط الأمساك والأنتظار

يمكن مهاجمة هذا الشرط بأجبار جميع العمليات علي طلب جميع الموارد التي سوف تحتاجها قبل بدء التنفيذ فإذا كانت كل الموارد متوفرة فسيتم تخصيص كل ما

أعداد :-

أ. الهادي سليمان الهادي　　جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

تحتاجه العمليه من موارد وتستطيع بالتالي التنفيذ حتي النهاية وأما اذا كان أحد الموارد أو أكثر غير متاحاً فلن يتم تخصيص أي مورد وبذلك سوف تضطر العملية للأنتظار . ولكن هذا الحل لا يمكن تنفيذه عملياً لأنه لا يمكن معرفه الموارد التي تحتاجها كل عمليه قبل بداية عملها ، كما أنه اذا كان الأمر كذلك فيمكننا أستخدام خوارزمية المصرفي بالأضافه الي انه توجد مشكله أخري وهي عدم أستغلال الموارد بشكل أمثل وعلي سبيل المثال اذا كان لدينا عمليه تقرأ البيانات من قرص ثم تحلله لمدة ساعه وبعد ذلك تقوم بكتابة النتائج في قرص أخر وتحتاج في عملها علي الماسحه فأذا أستوجب طلب كل الموارد مقدماً فأن العمليه ستحجز الماسحه لمدة ساعه دون أستخدامها .

٣. مهاجمة شرط عدم الأستيلاء

أن مهاجمة هذا الشرط أقل فائدة لأنه اذا أفترضنا أن لدينا عمليه تمتلك الطابعه وكانت في منتصف عملها ثم قمنا بإخذ الطابعه منها قسرياً وذلك لعدم وجود راسمه لأزمه لهذه العمليه يمكن أن يكون صعباً في أحسن الأحوال أو مستحيلاً في أسوئها .

٤. مهاجمة شرط الأنتظار الدائري

يمكننا مهاجمة هذا الشرط وذلك بعدة طرق ويمكن تمثيل أحدي الطرق ببساطه بوضع القاعدة التاليه

(لكل عملية يجب أن تستخدم مورداً واحداً فقط في أي لحظه ، فإذا أحتاجت مورداً أخر يجب عليها أن تحرر المورد الأول أولاً) .

ويمكن أن تكون هذه الطريقه غير مقبوله في بعض الأحيان فمثلاً أذا كانت لدينا عمليه تريد نسخ ملف ضخم من شريط الي الطابعه .

أعداد :-

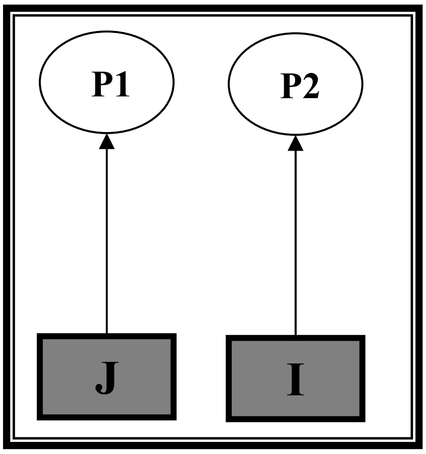
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب

مارس ٢٠٠٨

كما أنه توجد طريقه أخري لمهاجمة شرط الأنتظار الدائري وتتمثل في ترقيم جميع الموارد وتستطيع كل عمليه طلب الموارد كما تحلو لها ولكن بشرط أن تكون جميع الطلبات في تسلسل رقمي معين فمثلاً أذا كانت لدينا الموارد بالترقيم التالي

١. الماسحه . ٢. الطابعه . ٣. سواقه الشريط .

فإن أي عمليه يمكنها طلب الطابعه أولاً ثم سواقه الشريط ولكنها لا تستطيع طلب الطابعه أولاً ثم الماسحه وبذلك لا يمكن أن يحتوي غراف الموارد علي أي من الحلقات ، وعلي سبيل المثال أذا كان لدينا العمليتان P1 , P2 وتطلبان الموردين J , I علي التوالي وبإفتراض أن I , J هما موردان متمايزان (مختلفين) فأذا كان I > J هذا يعني أن العمليه P2 لا يمكنها طلب المورد J لانه أقل من المورد الذي بحوزتها حالياً ، فأما أذا كان I < J هذا يعني أن العمليه P1 لا يمكنها طلب المورد I لأنه أقل من المورد الذي بحوزتها حالياً ، وفي كلا الحالتين الأستعصاء مستحيلاً .



أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨

المراجع باللغة الأنجليزية : -

- Operating system concepts, Silberschatz & Galvin, Fifth edition 1999
- MODERN OPERATING SYSTEMS
  by Andrew S. Tanenbaum

المراجع باللغة العربية : -

- نظم تشغيل الحاسبات، د.مهندس محمد احمد فكرين، دار المريخ ١٩٩٦م
- ج آرتشر هاريس (ترجمة أمين أيوبي) أنظمة تشغيل الحاسوب، أكاديمياً، بيروت ٢٠٠٢م.

أعداد :-
أ. الهادي سليمان الهادي جامعة النيلين - كلية علوم الحاسوب - قسم علوم الحاسوب
مارس ٢٠٠٨